Forward Kinematics of Programmable Universal Manipulation Arm (PUMA) Robot Using Denavit-Hartenberg Representation

MECE 617 Activity

Ivan John A. Naparota
University of San Carlos – Talamban Campus
Department of Electrical and Electronics Engineering
ivanjohnnaparota66@gmail.com

***Abstract* — this activity presents the Forward Kinematics of a Programmable Universal Manipulation Arm (PUMA) Robot using Denavit – Hartenberg Representation and with the aid of a numerical computing software, MATLAB.**

Keywords – Forward Kinematics, Denavit-Hartenberg Representation, MATLAB

I. INTRODUCTION

Denavit–Hartenberg parameters (also called DH parameters) are the four parameters associated with a particular convention for attaching reference frames to the links of a spatial kinematic chain, or robot manipulator. Jacques Denavit and Richard Hartenberg introduced this convention in 1955 in order to standardize the coordinate frames for spatial linkages [2].

You just have to identify the parameters on each joint that is very needed in this convention, this will be defined further in the next parts of this paper. These parameters will then be used to perform the necessary operations that mainly involves multiplication of matrices for us to be able to obtain the position vector of the end effecter, which is the main concern of forward kinematics.

II. EQUATIONS AND MATLAB CODES

A. *Formulas and Equations*

For a specific *i*, we obtain the T matrix, $T = {}^{0}A_{i}$, which specifies the position and orientation of the endpoint of the manipulator with respect to the base coordinate system. Considering T matrix to be of the form [1]

$$T = \begin{bmatrix} cos\theta & -cos\alpha sin\theta & sin\alpha sin\theta & acos\theta \\ sin\theta & cos\alpha cos\theta & -sin\alpha cos\theta & asin\theta \\ 0 & sin\alpha & cos\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where:

d – depth along the previous joint's z axis

θ – angle about the previous z to align its x with the new origin

r – distance along the rotated x axis

α – rotation about the new x axis to put z in its desired orientation

See figure 2,

According to page 37 of the book "Robotics: Control, Sensing, Vision, and Intelligence" by Fu, Gonzalez, and Lee, established link coordinate systems for PUMA robot is hereby presented:

PUMA robot arm link coordinate parameters

Joint i	θ_i	α_i	a_i	d_i	Joint range
1		−90	0	0	−160 to +160
2		0	431.8 mm	149.09 mm	−225 to 45
3		90	−20.32 mm	0	−45 to 225
4		−90	0	433.07 mm	−110 to 170
5		90	0	0	−100 to 100
6		0	0	56.25 mm	−266 to 266

And θ's depend on the inputs of the joints that will dictate the position of the end effecter.

Therefore,

$$\text{Joint 1} = \begin{bmatrix} cos\theta & -\cos(-90)sin\theta & \sin(-90)sin\theta & 0*cos\theta \\ sin\theta & \cos(-90)cos\theta & -\sin(-90)cos\theta & 0*sin\theta \\ 0 & \sin(-90) & \cos(-90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Joint 2} = \begin{bmatrix} cos\theta & -\cos(0)sin\theta & \sin(0)sin\theta & 431.8 * cos\theta \\ sin\theta & \cos(0)cos\theta & -\sin(0)cos\theta & 431.8 * sin\theta \\ 0 & \sin(0) & \cos(0) & 149.09 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Joint 3} = \begin{bmatrix} cos\theta & -\cos(90)sin\theta & \sin(90)sin\theta & -20.32 * cos\theta \\ sin\theta & \cos(90)cos\theta & -\sin(90)cos\theta & -20.32 * sin\theta \\ 0 & \sin(90) & \cos(90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Joint 4} = \begin{bmatrix} cos\theta & -\cos(-90)sin\theta & \sin(-90)sin\theta & 0 * cos\theta \\ sin\theta & \cos(-90)cos\theta & -\sin(-90)cos\theta & 0 * sin\theta \\ 0 & \sin(-90) & \cos(-90) & 433.07 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Joint 5} = \begin{bmatrix} cos\theta & -\cos(90)sin\theta & \sin(90)sin\theta & 0 * cos\theta \\ sin\theta & \cos(90)cos\theta & -\sin(90)cos\theta & 0 * sin\theta \\ 0 & \sin(90) & \cos(90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Joint 6} = \begin{bmatrix} cos\theta & -\cos(0)sin\theta & \sin(0)sin\theta & 0 * cos\theta \\ sin\theta & \cos(0)cos\theta & -\sin(0)cos\theta & 0 * sin\theta \\ 0 & \sin(0) & \cos(0) & 56.25 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

And the homogeneous matrix ${}^0T_i$ which specifies the location of the *i*th coordinate frame with respect to the base coordinate system is the chain product of successive coordinate transformation matrices of ${}^{i-1}A_i$, and is expressed as [1]

$$ {}^0T_i = {}^0A_1\, {}^1A_2 \ldots {}^{i-1}A_i = \begin{bmatrix} 0R_i & 0P_i \\ 0 & 1 \end{bmatrix} $$

Where:

P – Position vector of the hand

B. *MATLAB Codes*

```
clc;
clear all
close all

%depth
d1 = 0;
d2 = 149.09;
d3 = 0;
d4 = 433.07;
d5 = 0;
d6 = 56.25;

%alpha
a1 = -90;
a2 = 0;
a3 = 90;
a4 = -90;
a5 = 90;
a6 = 0;

%length
l1 = 0;
l2 = 431.8;
l3 = -20.32;
l4 = 0;
l5 = 0;
l6 = 0;

%theta
theta1 = 'Provide Angle Theta for Joint 1: ';
theta2 = 'Provide Angle Theta for Joint 2: ';
theta3 = 'Provide Angle Theta for Joint 3: ';
theta4 = 'Provide Angle Theta for Joint 4: ';
theta5 = 'Provide Angle Theta for Joint 5: ';
theta6 = 'Provide Angle Theta for Joint 6: ';
t1 = input(theta1);
t2 = input(theta2);
t3 = input(theta3);
t4 = input(theta4);
t5 = input(theta5);
t6 = input(theta6);

%transformation matrices
transformation1 = [cosd(t1) -(cosd(a1)*sind(t1))
sind(a1)*sind(t1) l1*cosd(t1)
sind(t1) cosd(a1)*cosd(t1) -(sind(a1)*cosd(t1)) l1*sind(t1)
0 sind(a1) cosd(a1) d1
0 0 0 1];

transformation2 = [cosd(t2) -(cosd(a2)*sind(t2))
sind(a2)*sind(t2) l2*cosd(t2)
sind(t2) cosd(a2)*cosd(t2) -(sind(a2)*cosd(t2)) l2*sind(t2)
0 sind(a2) cosd(a2) d2
0 0 0 1];

transformation3 = [cosd(t3) -(cosd(a3)*sind(t3))
sind(a3)*sind(t3) l3*cosd(t3)
sind(t3) cosd(a3)*cosd(t3) -(sind(a3)*cosd(t3)) l3*sind(t3)
0 sind(a3) cosd(a3) d3
0 0 0 1];

transformation4 = [cosd(t4) -(cosd(a4)*sind(t4))
sind(a4)*sind(t4) l4*cosd(t4)
sind(t4) cosd(a4)*cosd(t4) -(sind(a4)*cosd(t4)) l4*sind(t4)
0 sind(a4) cosd(a4) d4
0 0 0 1];

transformation5 = [cosd(t5) -(cosd(a5)*sind(t5))
sind(a5)*sind(t5) l5*cosd(t5)
sind(t5) cosd(a5)*cosd(t5) -(sind(a5)*cosd(t5)) l5*sind(t5)
0 sind(a5) cosd(a5) d5
0 0 0 1];

transformation6 = [cosd(t6) -(cosd(a6)*sind(t6))
sind(a6)*sind(t6) l6*cosd(t6)
sind(t6) cosd(a6)*cosd(t6) -(sind(a6)*cosd(t6)) l6*sind(t6)
```

```
0 sind(a2) cosd(a2) d6
0 0 0 1];

%multiply
multiply1 = transformation1 * transformation2;
multiply2 = transformation1 * transformation2 *
transformation3;
multiply3 = transformation1 * transformation2 *
transformation3 * transformation4;
multiply4 = transformation1 * transformation2 *
transformation3 * transformation4 * transformation5;
multiply = transformation1 * transformation2 *
transformation3 * transformation4 * transformation5 *
transformation6;

%output
output = [multiply(1,4) multiply(2,4) multiply(3,4)]';
disp('End Effecter is Located on Point(x,y,z):');
disp(num2str(output,'%.2f'))

%plot
line([0 0],[0 0],[-660.4 0],'LineWidth',2)
line([0 multiply1(1,4)],[0 multiply1(2,4)],[0 0],'LineWidth',2,
'color','r');
line([multiply1(1,4) multiply2(1,4)],[multiply1(2,4)
multiply2(2,4)],[0 multiply2(3,4)],'LineWidth',2, 'color','g')
line([multiply2(1,4) multiply3(1,4)],[multiply2(2,4)
multiply3(2,4)],[multiply2(3,4) multiply3(3,4)],'LineWidth',2,
'color','y')
line([multiply3(1,4) multiply4(1,4)],[multiply3(2,4)
multiply4(2,4)],[multiply3(3,4) multiply4(3,4)],'LineWidth',2,
'color','b')
line([multiply4(1,4) multiply(1,4)],[multiply4(2,4)
multiply(2,4)],[multiply4(3,4) multiply(3,4)],'LineWidth',2,
'color','k')
axis([-1000 1000 -1000 1000 -1000 1000])
grid on

%message
disp('Type [PumaForwardDH] in Command Window to Try
Again');
```

IV. RESULTS AND DISCUSSIONS

As shown in Figure 1, the program will ask for the input thetas that will define the position of the end effecter of our PUMA robot, thetas are the only inputs we have because the other parameters are already defined as shown above.

For these given thetas,

Theta for Joint 1: 90

Theta for Joint 2: 0

Theta for Joint 3: 90

Theta for Joint 4: 0

Theta for Joint 5: 0

Theta for Joint 6: 0

And considering the defined parameters presented above,

End effecter is located in (x,y,z) point (-149.09, 921.12, 20.32).

To verify if the presented end effecter of the program is correct, refer to page 45 of the book “Robotics: Control, Sensing, Vision, and Intelligence” by Fu, Gonzalez, and Lee,

“As a check, if ϑ_1 =90', ϑ_2 =0', ϑ_3 =90', ϑ_4 = 00, ϑ_5 =0', ϑ_6 =0', then the T matrix is”

$$\mathbf{T} = \begin{bmatrix} 0 & -1 & 0 & -149.09 \\ 0 & 0 & 1 & 921.12 \\ -1 & 0 & 0 & 20.32 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This validates the end effecter presented by the program.

V. CONCLUSION

Denavit – Hartenberg representation is capable of solving the forward kinematics of Programmable Universal Manipulation Arm (PUMA) robot, you just have to define the parameters on each joint of the PUMA robot and do the necessary operations that mainly involve multiplication of matrices.

Also, MATLAB’s computing prowess would help a lot to perform the necessary operations without the need for manual solutions.

VI. REFERENCES

1. Fu, K. S., Gonzales, R. C., .Lee, C. S. G., ROBOTICS : Control, Sensing, Vision, and Intelligence.
2. https://en.wikipedia.org/wiki/Denavit%E2%80%93Hartenberg_parameters

VII. FIGURES

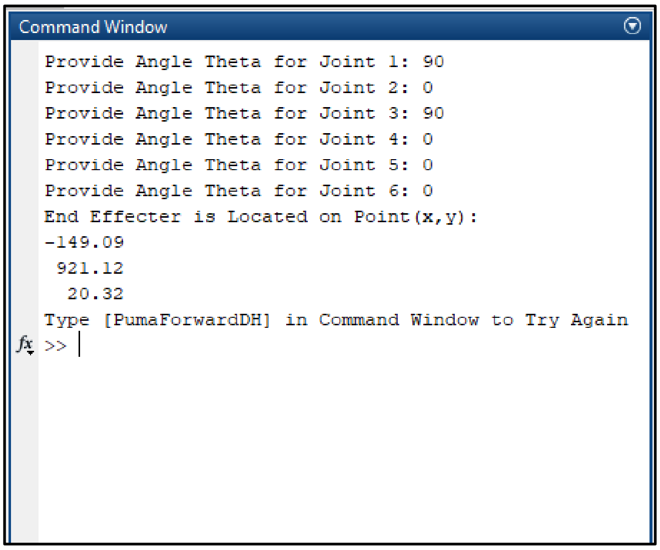
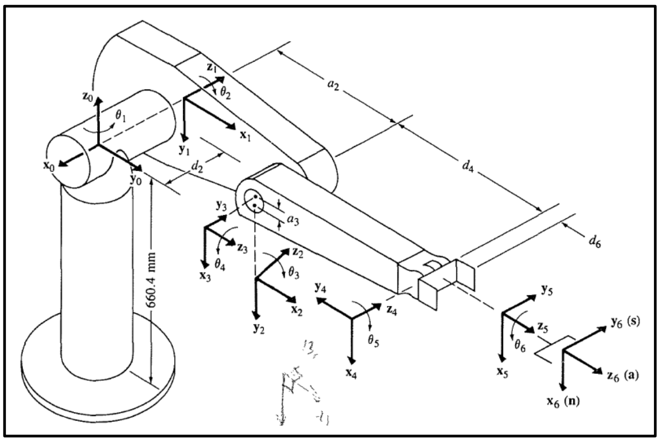


Figure 1MATLAB Command Window



Fu, K. S., Gonzales, R. C., .Lee, C. S. G., ROBOTICS : Control, Sensing, Vision, and Intelligence.

Figure 2 PUMA Robot